UNCLASSIFIED

DTIC FILE COPY

(1)

AD-A206 146

REPORT DOCUMENTATION PAGE — Form Approved OMB No. 0704-0188

1b. RESTRICTIVE MARKINGS

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release.
Distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
MAG 87-011-01

5. MONITORING ORGANIZATION REPORT NUMBER(S)
MAG 87-011-01

6a. NAME OF PERFORMING ORGANIZATION
Software Production Branch

6b. OFFICE SYMBOL (If applicable)
WR-ALC/MAITC

7a. NAME OF MONITORING ORGANIZATION
MATE Application Group

6c. ADDRESS (City, State, and ZIP Code)
Robins AFB GA 31098-5149

7b. ADDRESS (City, State, and ZIP Code)
WR-ALC/MAIT
Robins AFB GA 31098-5149

8a. NAME OF FUNDING / SPONSORING ORGANIZATION

8b. OFFICE SYMBOL (If applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS (City, State, and ZIP Code)

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
1750A, MOS & MATE

12. PERSONAL AUTHOR(S)
Donald B. McComb

13a. TYPE OF REPORT
Final

13b. TIME COVERED
FROM ______ TO ______

14. DATE OF REPORT (Year, Month, Day)
88 MAR 09

15. PAGE COUNT
21

16. SUPPLEMENTARY NOTATION
— fr. p. ii

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The paper argues that MIL-STD-1750A computers are not viable for use in Modular Automatic Test Equipment (MATE), and that both the 1750A and the MATE Operating System (MOS) should be dropped as MATE Program standards.

The paper addresses both technical and economic reasons. (SDW)

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION

22a. NAME OF RESPONSIBLE INDIVIDUAL
LEROY HUDSON

22b. TELEPHONE (Include Area Code)
912-926-1501

22c. OFFICE SYMBOL
MAITBD

Report MAG 87-011-1

1750A, MOS & MATE

Authors:

Donald B. McComb
WR-ALC/MAITC
Warner Robins AFB, Ga 31098

Walter J. Marusa Jr.
SA-ALC/MATTS
Kelly AFB, Tx,

9 Mar 1988

FINAL REPORT

Prepared for:

MATE Applications Group
WR-ALC/MAIT
Robins AFB 31098-5148

APPROVAL SHEET

Report ID: MAG 87-011-01

Title: 1750A, MOS & MATE

Author(s): Donald B McComb Date: 9 Mar 88

Walter J Mor Jr. Date: 9 MAR 88

Date:

Standing Committee Chairperson: C. J. Smith Jr. Date: 9 Mar 88

Executive Committee Chairperson: Larry K. Israel Date: 6 May 88

Preface

This report was written by members of the MATE Applications Group (MAG) to assist the Modular Automatic Test Epuipment (MATE) program.

MATE is an Acquisition Management program established by AFSC/AFLC regulation 800-23.

The MAG was established in recognition of the need for Air Force users to influence the application and future of MATE. This need was identified during the AFLC MATE Conference held at Wright Patterson AFB on 31 March 1987.

The ~~overall~~ objective of the MAG is to support and improve the MATE concept and programs. This will be accomplished by assessing the needs of the maintenance community and establishing a means of communicating those needs from the operating/user (maintenance) organizations to the managing/acquisition organizations. 1473

This report must be considered in the context of the MATE program.

Acknowledgements

We would like to acknowledge the help of:

Tom Flores
Scott Halwes
Michael McGlothing
William Muellar
Carlos Vera

All of:
SA-ALC/MATT
Kelly AFB, Tx. 78241
(512) 925-6557
AV 945-6557

James Goddman
OO-ALC/MAKEEF

O.T. Smith
WR-ALC/MAIT

TABLE OF CONTENTS

1 ISSUE

Is the MATE requirement for a standard computer architecture (MIL STD 1750A) and a standard operating system (MOS) viable in current and future MATE implementations? The original MAGIT is included in Appendix A.

1.1 EXECUTIVE SUMMARY

The following paper shows quite clearly that:

a. Current testing requirements exceed the capabilities of 1750A and MOS.

b. The 1750A and MOS are not required to meet MATE's overall objectives.

The recommended actions to correct the problem are:

a. The MATE Program should eliminate the requirement for the 1750A architecture and update AFSC/AFLC 800-23 appropriately.

b. The MATE Program should eliminate the requirement for the MATE Operating System in favor of a User Shell (ref. para 3.3.6) and update AFSC/AFLC Reg. 800-23 appropriately.

c. The MATE Program should promote the conversion of the MATE Control Support Software (MCSS) to Ada.

2 FACTORS BEARING ON THE ISSUE

2.1 Background

The 1750A computer architecture was chosen for use in MATE ATE about 10 years ago. At that time it was a good choice. Primary benefits included:

The capability to address 2 Megabytes of main memory.

A Standard Architecture would allow a standard operating system. This supports application software portability and MATE goals of ATE standardization.

It was state of the art.

Since it was an established MIL STD, computer manufacturers could choose whether or not to invest time and money into a share of the Air Force market for ATE computers. This would result in both simpler and fairer competition.

These benefits outweighed the disadvantages of the 1750A architecture. They were:

> The 1750A is optimized for avionics applications. However, ATE requires considerable character manipulation and the 1750A does not excel in this area.
>
> The 1750A standardizes the CPU architecture, but does not standardize the I/O implementation.

These limitations were accepted because there was no viable second choice and the 1750A became the MATE computer architecture. The MATE Operating System, or MOS, was then developed with the intent that it would be re-hostable on the expected flood of MATE computers.

2.2 Definitions

Ada - Is a MIL-STD computer language (ANSI/MIL-STD-1815A), it is the first truly designed computer language.

AIL - ATLAS Intermediate Language is an intermediate language between source (ATLAS) and executable code. The MTE interprets AIL at runtime to create executable code.

ATE - Automatic Test Equipment, a collection of test instruments controlled by a computer.

ATS - Automatic Test System is an ATE with the addition of the TPSs.

bit - Binary digit, either 0 or 1. Also the smallest storage location.

byte - A group of 8 adjacent binary digits that a computer processes as a unit.

C - A computer language designed to be used with UNIX.

CIIL - Control Interface Intermediate Language. The language used to "talk" to a MATE instrument over the IEEE 488 bus.

Computer System - Consists of both the CPU hardware and the operating system software.

I/O - Input/Output

JOVIAL - A MIL-STD computer language designed for use in an avionics environment.

MATE - Modular Automatic Test Equipment. As used in this document it refers to the hardware/software specified as "MATE".

MAC - MATE ATLAS Compiler. Its main purpose is to compile ATLAS source code to AIL.

MCSS - MATE Control Support Software. Consists of MAC, MOLE, MTE and MOS (defined elsewhere).

MOLE - MATE On-Line Editor is an editor used to generate ATLAS source code in the MATE environment.

MOS - MATE Operating System is the 1750A operating system used in MATE.

MTE - MATE Test Executive is what actually controls the ATS during a test. It takes the AIL, interprets it into CIIL, and executes it.

RAM - Random Access Memory is memory that makes stored data immediately available when addressed regardless of previous address. The data in this memory can be changed.

ROM - Read Only Memory is similar to a RAM, however, its data can't be changed.

TPS - Test Program Set is used with an ATE to test a UUT. A TPS consists of an Interface Test Adapter(ITA), a software program and Technical Orders (TOs).

TRU - Tester Replaceable Unit, a controller, instrument, power supply, etc. Is a discrete module which can be used in different testers.

User Shell- Is an extra set of commands added to a computer to make it resemble another computer.

Whetstones- A standard series of tests used to compare the performance of different computers.

word - Is the amount of memory a CPU can perform an operation on or address at one time. Usually a multiple of 8 bits or a byte. As the 1750A is a 16 bit machine its word contains 16 bits, or two bytes. Note: A 32 bit machine's word is 4 bytes an 8 bit machine's word is 1 byte etc.

2.3 Facts

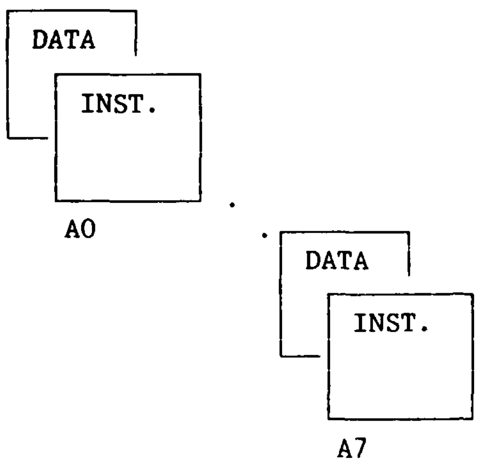
2.3.1 1750 Background

MIL-STD-1750A defines the Instruction Set Architecture (ISA) for a 16 bit airborne computer. The standard was released 2 July 1980 (1750 was released 21 Feb 1979). A further upgrade (1750B) is currently in work, but no firm release date is available.

2.3.2 Memory

The 1750A is a 16 bit machine and uses 16 bit words to address 2 bytes at a time. It also uses 16 bits for addressing. Since 16 bits can only address 64K words directly it uses a paging and bank address state access scheme to increase the memory it can address. This results in the 1750A being able to address a respectable amount of memory, 2 Megabytes or 1 Megaword as 2 bytes make a word.

Memory is divided into 8 address states (128K words each). Only one is accessible at a time. Each address state (A0-A7) is divided into 32 pages of 4k words of memory. Half is used for Data and half for Instructions.



Paging Scheme

This results in 128K words being available for use at any time. To address other states requires additional overhead ,thereby affecting performance.

2.3.3 Instruction Set

The basic 1750 was designed many years ago for number crunching required in avionics. Because of this, it lacks some of the architecture and instructions needed to manipulate character data. This causes inefficiencies when doing a lot of character manipulation as required when compiling and interpreting TPSs.

Character data is stored in 8 bit ASCII codes.

Example: 1000100 is the character D

Typically if a computer is a 16 bit machine two characters are stored in each 16 bit word.

Example:

1001110	1000001	Stores 2 characters "AN"
15	0	

Because the 1750A is missing the instructions required to manipulate both characters stored in a word the data must instead be stored in two words as shown below:

0000000	1000001	Character "A"
0000000	1001110	Character "N"
15	0	

Not only does this waste memory but it also increases overhead if the data is packed and unpacked as it is moved into and out of mass storage devices (disc drives, tape drives, etc.).

2.3.4 **Microcode**

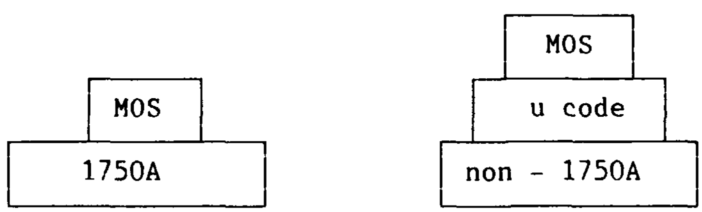
Computers built from the ground up to be 1750s have historically been very expensive. This is because as airborne units they have been also designed and built to stricter standards than needed for ground applications such as ATSs.

To our knowledge only three 1750A style computers have been used in MATE stations:

UNIVAC 1630 - Native 1750A, expensive, and out of production
HP A700 - Hewlett-Packard microcoded to 1750A
HP A900 - Hewlett-Packard microcoded to 1750A

Two of the computers achieved 1750A qualified status via emulation in microcode. This reduces hardware costs but it adds to software costs since microcode is software implemented as firmware. It contains programs resident on a hardware Integrated Circuit(ROM), rather than in memory(RAM) or on disk

Because emulation adds an extra layer of software between the operating system and the host CPU (ref diagram below) it requires more computational (compute) cycles to execute any given command. Thus, the performance will be severely degraded (between a factor of 10-20) over running the CPU in native mode.



Native 1750A versus microcoded 1750A

The fact emulation is firmware makes it subject to having errors (bugs) that have not been identified or resolved. Reference MAGIT 88-001 written against the A900 computer that had passed SEAFAC testing.

2.3.5 Performance

SA-ALC did a study of computer computational capabilities using the Lawrence Livermore Laboratories modification of the Whetstone (B1) Benchmark. This study was done in support of another MAGIT 87-035, Non-Electronic Testing. The entire study is available upon request.

Although there are many programs which could be used as a benchmark, this Whetstone was chosen due to its thorough use of computer operations and also because of the acceptance of the Whetstone as an industry standard. The Whetstone program consists of different sections or modules each of which performs a different type of operation a predefined number of times. The module operations consist of simple identifier calculations, operations on array elements, subroutine linkage with arrays as parameters, conditional jumps, integer arithmetic, trigonometric functions, procedure calls, array references, and the use of standard functions (i.e. square root, logarithm, etc.). One Whetstone is defined as the execution of all of the modules' iterations.

The computer systems used in the study were an IBM PC AT, IBM PC XT, PC 80386, VAX 8300, HP1000 A900, HP1000 A700, MATE 1750A A900, MATE 1750A A700, and the UNIVAC 1630. The languages used were Ada, ATLAS, FORTRAN and JOVIAL. This was necessitated by not having the same compilers on each computer, where more than one compiler existed for a system, multiple tests were done. All test cases were done based on the time to execute 257 million Whetstones and the costs were based on the typical system cost. Some data has been extracted and is presented below.

Price versus Performance

COMPUTER	LANGUAGE	COST	WHETSTONES/SEC
A700 (1750A)	JOVIAL	$70,000	30228
A900 (1750A)	JOVIAL	$80,000	46024
UNIVAC 1630	JOVIAL	$100,000	72598
IBM PC	FORTRAN	$2,000	74633

2.3.6 MAC, MOLE & MTE

MAC, MOLE and MTE are application programs as well as the critical standards in MATE Control and Support Software. MOLE (MATE On Line Editor) is used to create a file containing an ATLAS source program. MAC (Mate ATLAS Compiler) takes these files and creates an AIL (ATLAS Intermediate Language) file. MTE (MATE Test Executive) then uses the AIL to create the CIIL (Control Interface Intermediate Language) as well as control the execution of tests.

2.3.7 MOS

The MATE Operating System was derived from a version of UNIX (trademark of Bell Labs). The version, of UNIX it was derived from, was public domain and not very competent. It was then translated from C to JOVIAL using a translator. The use of a translator further degrades performance since optimal code is rarely achieved.

Between May of 1984 and June of 1987, 163 Software Trouble reports were written against the MOS. This compares to 253 written against the MAC, MOLE, and MTE combined. Of the 70 STRs written specifically against the 4.0 MOS, 45, or over 50%, were fixed in 5.0, the next version. On the surface, this seems acceptable until one looks at ongoing programs such as DATSA. There is a reluctance on the part of program managers to update to a later operating system due to 5.0 not being upward compatible and the potential of having to redo TPSs. If DATSA does not upgrade, future TPS development efforts will have to make do with work-arounds and temporary fixes. This is not just a DATSA problem. The tendency is to "fix" problems by putting the correction in the next block upgrade. This leaves users with the choice of living with what they've got or trying to upgrade and suffer possible compatibility problems.

Commercial operating systems, with their extremely large user base, do not have the level of problems exhibited by MOS and the corresponding correction costs. Commercial operating systems, must be clean products, and provide upward compatibility or their vendors will quickly go out of business.

2.4 Assumptions

a. It is assumed that the decision whether to retain the 1750A/MOS is to be based on factual data and not on unsupported opinions.

b. It is assumed that a standard must provide a substantial benefit.

c. Most of all it is assumed that the underlying purpose of MATE is to do the best job for the lowest possible life-cycle-cost.

2.5 Criteria

The requirements for a MATE computer system today are fundamentally the same as 10 years ago. The computer must be:

a. Economical to acquire.

b. Supportable over a reasonable lifetime.

c. Capable of meeting the required performance standards.

d. Compatible with use of baselined MAC, MOLE and MTE. With only minor, if any, modifications. In other words it must support the portability of MAC, MOLE and MTE thus, the portability of TPSs and TRUs.

e. Usable, with minimal training. By having very similar operator command interface training costs will be reduced considerably.

f. Replaceable at minimal cost.

3 DISCUSSION

3.1 Purpose of the 1750A/MOS

In the MATE world MOS and the 1750A exist to run the rest of the MCSS. Thus if MAC, MOLE, and MTE could run on other computers and operating systems, without impacting supportability and portability, the decision on what computer system to use would be based on price and performance.

It is our goal to show that the 1750A and MOS are not necessary to meet the criteria stated in paragraph 2.5.

3.2 Limitations of the 1750A/MOS

3.2.1 1750A architecture:

3.2.1.1 Memory

Today's TPSs require large amounts of memory to both compile and run efficiently. This requirement exceeds the 2 megabytes of memory the 1750A supports. In addition the 1750A's memory addressing scheme is inefficient and causes compilers problems. In fact the "Ada Adoption Handbook A Program Manager's Guide" lists the use of Ada with the 1750A as "Applicable with Reservations" (Technical Report CMU/SEI-87-TR-9 May 87 page 42).

3.2.1.2 Instruction Set Architecture

Character manipulation is important in all three MCSS application programs MAC, MOLE and MTE. Compilation of any language requires a lot of character manipulation because a source file is always 100% ASCII. It is even more important for MATE ATLAS since the result after compilation is ATLAS Intermediate Language which is mostly ASCII characters. An editor, such as MOLE, deals almost exclusively with ASCII characters. And since MTE uses AIL and CIIL it also requires a lot of character manipulation. Yet the 1750A is very poor at character manipulation(ref para 2.3.3).

3.2.1.3 Other Architectures in Avionics

The state-of-the-art has come a long way since the 1750 was designed. There are a lot of good machines out there. In fact many weapon systems are using computers other than the 1750A.

WEAPON SYSTEM CPUS

PLATFORM	SYSTEMS	CPU(S)
F-4G	APR-38	TI (CUSTOM)
	APR-47	UNISYS 1630 (1750)
F-15	AN/ALE-45	MOTOROLA 6800
B-1B	ALQ-161	8086
		EATON/AIL (CUSTOM)
F/EF-111	ALR-62I	2 FAIRCHILD 9445s
		7 INTEL 8051s

3.2.1.4 **Microcode**

A common method used to achieve a 1750A architecture in MATE is to microcode commercial computers. While in theory this may look good, there are problems in the application of the concept:

a. A microcoded 1750A computer is 3-40 times more expensive than commercial computers of the same power.

b. In addition since microcode is software (firmware) it introduces more opportunities for mistakes. Additional manpower and funding resources are required to control and support the microcode. And since the microcode is like a foundation, if the microcode is incorrect there will be all sorts of problems.

c. Performance is severely degraded when a CPU is microcoded to emulate a different machine. The table below is extracted from the San Antonio study mentioned previously(ref para 2.3.5).

PERFORMANCE MICROCODE VS NATIVE MODE

COMPUTER	WHETSTONES/SEC
1750A Microcode A900 (JOVIAL)	46,024
Native A900 (Ada)	736,384
Native A900 (JOVIAL)*	-------
Native A900 (Ada (nocheck))	1,011,811
Native A900 (FORTRAN)	1,586,419

* Est. of where JOVIAL in a native A900 would have placed if available for testing. Based on Ada and JOVIAL test results on a VAX 8300.

As can be seen there is a performance loss of at least a factor of 16 between a 1750A microcoded A900 and a native mode A900.

3.2.2 **MATE Operating System (MOS)**

Operating Systems make or break a computer application. Any errors make it very hard on application programs. Put an application on a poor operating system the result is like building a house on sand dunes, it will be stable only until the first big wave. A good clean operating system supports the operator. And will handle any mistakes made, by the operators or application programs, and handle the errors without the system crashing.

3.2.2.1 **Cost**

An operating system is difficult to write and requires extensive use to iron out all the bugs, if ever. This results in a very expensive piece of software that needs many users to amortize its development and support costs.

3.2.2.2 **Stability**

As pointed out earlier MOS has many problems. We believe the small user base and funding constraints, of USAF applications, will make it very difficult to evolve MOS to a competent, much less state-of-the-art, operating system.

3.2.3 **The combination of 1750A & MOS**

3.2.3.1 **Performance**

It is SLOW. It is painful for a user to sit down and execute a Test Program on a system that does not match the performance of the system it replaced. An example of this is the GPATS DATSA replacement.

While the performance can be improved, the Whetstone study SA-ALC did shows that much better performance can be achieved at MUCH less cost by using commercial computers. Even 1750As running at high clock rates will be limited by their architecture.

Most efforts to improve the performance of the 1750A are looking at 10-40 percent improvements. But, our testing needs require an increase in performance between a factor of 10 and 100.

It is of special significance that even the fastest 1750A available was beaten by a Personal Computer (PC).

3.2.3.2 **Portability**

Application portability is now very common across operating systems as long as they share a common language. Jovial is not a widely used language, but Ada is and could be used. And there are already plans under way in the MATE world to make the conversion. As many Ada compilers are being developed, portability should be excellent.

In addition, if MOS is dropped the MCSS conversion to Ada would be simplified significantly as MOS, about 25% of the code in MCSS, would not need to be converted.

3.3 Potential advantages of 1750/MOS

Assessment of options available has generated some common arguments against dropping the 1750A/MOS requirement. The following paragraphs address these arguments.

3.3.1 A commercial computer cannot run MAC,MOLE & MTE.

Not true, these programs are written in JOVIAL and can be executed on any computer that hosts the J73 compiler. At this time two applications of this have been accomplished:

a) The LANTIRN Intermediate Automatic Test Equipment (LIATE) being built by Martin Marietta.

b) AIS of New York announced the availability of MCSS version 5.0 on an IBM PC running Unix on December 1, 1987.

The main restriction now is that MAC, MOLE & MTE are written in JOVIAL and JOVIAL compilers are not widely available. However, since plans are already laid to rewrite the MCSS in Ada, the compiler problem should disappear. Thus while our choice of CPUs is currently somewhat restricted, when rewritten in Ada almost any CPU will be able to host MAC, MOLE & MTE.

3.3.2 Commercial computers cannot be maintained for 20+ years.

Since most MATE 1750As are commercial computers that have been modified (micro coded) this does not seem a valid argument. If an A700 or A900 with a microcode card in them can be supported why can't they be supported without the microcode card?

The only native 1750A computer used in a MATE system is the UNIVAC 1630 and it is out of production. This raises serious questions about its supportability.

3.3.3 Instrument transportability will suffer.

Transportability as a goal of MATE will not be impaired, since the computer need only tie into the IEEE 488 bus to run a MATE station. The ATE instruments respond to CIIL, and the means used to generate CIIL are of no consequence. Swapping computers will be no more difficult than it is now.

3.3.4 Commercial operating systems can't be supported for 20+ years

This is also a non-problem. Most operating systems in ATE are baselined and never upgraded for fear of impacting the TPSs. Only severe flaws force the upgrading of operating systems. This is much less likely in commercial operating systems, because of the large user base, than in MOS. In addition, the vast majority of commercial manufacturers maintain upward compatibility or provide a means to upgrade application software as they release new versions of their operating systems.

3.3.5 **The MCSS must be transportable.**

Today, many software packages are transported between systems without a common operating system. The major limitation at the moment is the lack of JOVIAL compilers. To increase portability, MAC, MOLE and MTE should be rewritten in Ada.

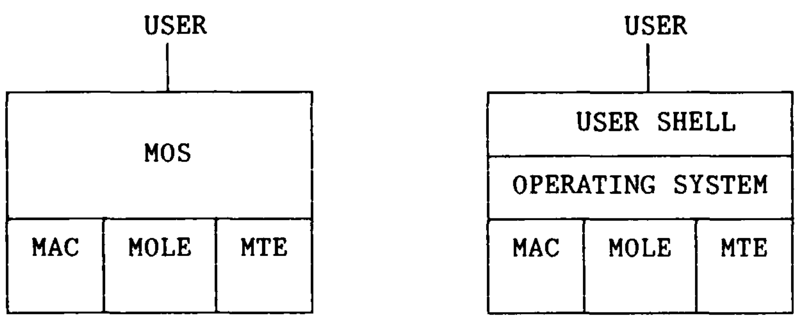
At least two commercial operating systems, hosts JOVIAL compilers, and run MAC, MOLE and MTE (the DEC VMS systems and the AIS micro ref. para. 3.3.1 above). DEC VMS hosts the MATE High Volume TPS Set and is becoming the standard MATE Program Development Station. The LANTIRN program is using a MicroVAX instead of a 1750A machine to host MAC, MOLE and MTE.

So transportability is acheivable without the 1750A and MOS.

3.3.6 **Different computer systems will increase training costs.**

If these programs (MCSS) ran on another computer system the only visible difference would be in the user interface (what the operator would see prior to invoking MAC, MOLE or MTE). In the Test Program development environment, only 5% of the developers time is spent interfacing with the operating system. In the shop environment, the interface to the operating system is even less. The production operator typically utilizes the operating system only to load a program.

Should a standard interface be desired, a MOS Command shell may be created for the commercial computers. This shell lays on top of an operating system and provides a common command structure. Maintaining/supporting the commercial shell would entail a much smaller cost then either; microcoding/MOS or MOS alone.



MOS versus Commercial Operating System & User Shell

	1750A	COMMERCIAL
MICROCODE	MAYBE	NO
MOS	YES	NO
MAC	YES	YES
MOLE	YES	YES
MTE	YES	YES
ATLAS	YES	YES
CIIL	YES	YES
MOS SHELL	NO	YES

1750A versus Commercial CPU

The table above summarizes the differences between the two approaches. MOS is only used with the 1750A, but MAC, MOLE, MTE and CIIL are used in both approaches. The MOS shell would have to be written but would only be about 5% the length of MOS, and could be done by either the computer manufacturer or the integrator since it is simpler than an operating system. This shell would allow user transition from MATE station to MATE station by providing MOS's command structure no matter what operating system is used, while still allowing the developer to interface with the "native" operating system.

Appendix B contains a proposed MOS Shell. At its simplest, the shell need only contain those commands needed by an operator to interface to the ATS for TPS loading and execution. The TPS developer could make use of all features of the native operating system for development purposes. This would create the least impact on Computer manufacturers, yet maintain the standardization desired by MATE.

3.3.7 Proliferation of OS increases MCSS support.

This is also invalid. By eliminating MOS the size of the mcss will be reduced by approximately 25%. as noted above, the commercial system will require minimal maintenance after it is fielded, and the shell, if required, would be easy to maintain.

4 CONCLUSION

The purpose of the 1750A and MOS was to insure the transportability of MAC, MOLE, MTE, TRUs and TPSs. But transportability can be achieved without 1750/MOS, indeed 1750/MOS impede more than abet transportability. Therefore, the 1750A and MOS should be dropped as requirements of the MATE program. (Although the 1750 or any MIL-STD computer could be used if it makes sense.)

The benefits to the realization of the MATE goals are:

1) Reduce life cycle costs

a. Reduced computer acquisition costs. Basic ATE systems could be run with PC's costing thousands of dollars, rather than specialized architecture computers costing tens of thousands of dollars.

b. The amount of software needing support in the MCSS would be reduced by approximately 20%. More in cases where microcode is eliminated.

c. The non-recurring cost of replacing a computer will be about the same whether a 1750A or commercial computer is used. However, the recurring costs will be much lower.

d. The SEAFAC requirement and its inherent delays and cost would be eliminated.

e. CPU acquisition time would be reduced. Current lead times for a 1750A computer can be several months to a year.

2) Reduce proliferation

a. MATE would be better able to compete with commercial systems in the areas of high speed digital testing and real-time testing. Both of these areas must be realistically excluded from current MATE standard applications because of CPU limitations.

b. In general, todays micro and mini computers are suited to a much wider range of tasks than the 1750A. The transition to portable testers, instruments-on-a-card, and other advanced technology would be eased by the elimination of the 1750A/MOS requirement. CPU's could be better matched to the task. A program office could select the CPU power required. For example, a single board CPU for a portable tester or a 32 bit CPU for a large EW tester.

3) Foster competition

a. An obvious benefit. By opening the market to more manufacturers rather than the handful today, the credibility of the MATE program would be enhanced.

b. More manufacturers will result in a more competitive atmosphere thereby reducing costs to the Air Force.

5 ACTION RECOMMENDED

1. The MATE Program Policy control points should expand the options allowed for MATE ATS computers and update AFSC/AFLC Reg. 800-23 appropriately.

2. The MATE Program Policy control points should eliminate the requirement for the MATE Operating System (MOS) in favor of a user shell and update AFSC/AFLC Reg. 800-23 appropriately.

3. The MATE Program Office should continue to promote the conversion of the MCSS to Ada.

APPENDIX A

MATE APPLICATION GROUP INITIATIVE TRAVELER

MAGIT # 87-011

SECTION A REQUESTOR INFORMATION

NAME: Donald McComb **DATE:** 06 AUG 87
ORGANIZATION: WR-ALC/MAITC **PHONE:** AV 468-5061
ADDRESS: ROBINS AFB GA 31098 **DDN ADDRESS:** ______

SECTION B REQUEST

TITLE: 1750A, MOS and MATE

CATEGORY: X **HARDWARE** X **SOFTWARE** ___ **NEW TECHNOLOGY** ___ **TOOLS**
___ **DOCUMENTATION** X **STANDARDS/GUIDES** ___ **REGULATIONS**

SUMMARY: The current standards are limiting the full capabilities available in todays computers. It is time to look at changing those standards.

IMPACT: The 1750A and MOS are seriously restricting MATE tester capabilities.

SUGGESTED IMPROVEMENT: Study future course of testing technology and recommend new things we want to standardize on.

SECTION C EXECUTIVE COMMITTEE DISPOSITION

DATE ______ **APPROVED** ___ **DISAPPROVED** ___ **DEFERRED** ___
ASSIGNED TO TECHNICAL SC **DUE DATE** ______
REMARKS: Investigate more , combine with MAGIT's 87-005,87-015, and 87-030.

SECTION D FOLLOW-UP

DATE ______ **ACTION TAKEN:**

DATE ______ **ACTION TAKEN:**

DATE ______ **ACTION TAKEN:**

DATE ______ **ACTION TAKEN:**

SECTION E CLOSE-OUT

ACTION TAKEN:
APPROVALS:

APPENDIX B

MOS SHELL

1.0 Existing MOS commands extracted from MOS User's Manual

System Utilities

Write all Buffers to disc	sync
Mount/Unmount a File System	mount
Print or Set Date	date
Post Mortem Dump	pmd
Task Termination	kill
Print Process Status	ps
Indicate who is on the System	who
Find Free Space Left in a File System	df
Set Terminal Attributes	setcrt
Send Data Over IEEE Bus	bustalk
Display Device Errors	dsperr

File Utilities

Make a Directory	mkdir
List a Directory	ls
Move a File	mv
Copy a file	cp
Compare Two Files for Equality	cmp
Concatenate Files	cat
Print a File (Formated)	pr
Remove a File	rm
Change File Mode	chmod
Dump a file in hex or ascii	od
Link a File to a new name	ln
Read or Write File(s) to Tape	tp
Change Directory to a Specified Directory	cd
Print Current Directory Pathname	pwd
Change Ownership of a File	chown

Maintenance Utilities

Initialize a File System	mkfs
Dump a File System to Tape	dump
Restore a File System from Tape	restor
Add a User Account	addact
Make a Device File	mkdev
File System Check	fsck
Power Control Monitor Status	statpcm
Change Login Password	passwd

APPENDIX B

2.0 MOS commands recommended for inclusion in user shell.

System Utilities

Print or Set Date	date
Task Termination	kill
Print Process Status	ps
Indicate who is on the System	who
Find Free Space Left in a File System	df
Send Data Over IEEE Bus	bustalk

File Utilities

Make a Directory	mkdir
List a Dircctory	ls
Move a File	mv
Copy a file	cp
Compare Two Files for Equality	cmp
Print a File (Formated)	pr
Remove a File	rm
Read or Write File(s) to Tape	tp
Change Directory to a Specified Directory	cd
Print Current Directory Pathname	pwd

Maintenance Utilities

Power Control Monitor Status	statpcm